Advanced Robotics, Vol. 20, No. 4, pp. 391–412 (2006)

Also available online - www.vsppub.com

Full paper

Reasoning of abstract motion of a target object through task order with natural language — pre-knowledge of object-handling-task programming for a service robot

RIE KATSUKI [1,*], ROLAND SIEGWART [2], JUN OTA [1] and TAMIO ARAI [1]

[1] *The University of Tokyo, Tokyo 113-8656, Japan*
[2] *Swiss Federal Institute of Technology Lausanne (EPFL), CH-1015 Lausanne, Switzerland*

Received 26 November 2004; accepted 26 March 2005

Abstract—In this paper, an attempt is made to reason an abstract motion of a target object when the object is manipulated through a task order. The reasoning algorithm is used as a function in the system that navigates layman's robot programming. The task order consists of two words: Task and Target (e.g., 'switch on' and 'light'). There are three kinds of motions: Linear, Circular and Point-To-Point motion. The system chooses a suitable motion. In the reasoning, it is important to be able to reason from various input words using a limited knowledge base. Therefore, a knowledge base is proposed that consists of a thesaurus and minimum knowledge. The knowledge defines only words that directly stand for the motions (e.g., 'turn' means Circular motion). The knowledge is propagated through hypernyms and hyponyms in the thesaurus. A motion is reasoned using the propagated knowledge in Task and Target. Moreover, learning, which results from on-site updating of the knowledge from the user, achieves reinforcement/customization of the knowledge base. The system successfully reasoned motions from various task orders. Moreover, for the robot programming of a door-opening task, a robot with the reasoning system reasoned a motion of the door and realized the task.

Keywords: Reasoning; semantics; thesaurus; robot programming; service robot.

1. INTRODUCTION

Realization of various object-handling tasks by a service robot has been expected for a long time. Since it is currently difficult for a robot to acquire a task autonomously, programming by a user is a feasible approach. In particular, teaching-by-showing (programming-by-demonstration) is a powerful method of robot programming. In teaching-by-showing, a system extracts pre-defined task features from its visual

*To whom correspondence should be addressed. E-mail: rie.katsuki@toshiba.co.jp

observation of the user's demonstration; therefore, even laymen can easily program the task.

Many kinds of task features have been proposed in past studies. Kuniyoshi *et al.* [1] have extracted hand–object actions by qualitative changes in camera images (e.g. an object has disappeared, which means an object was picked up). Maeda *et al.* [2] have extracted pushing motion and pick-and-place motion by three-dimensional Hough transformation. Takamatsu *et al.* [3] have extracted transitions of contact configurations between two blocks. Billard *et al.* [4] have applied a probabilistic analysis to data in Cartesian and joint spaces to classify five manipulation strategies (e.g., moves only a specific box or moves all boxes in a specific direction). However, it is difficult to extract the task features correctly because a human demonstration, which is intrinsically not mechanical, always includes errors.

In this study, therefore, an attempt is made to extract a feature based on a linguistic interpretation of a task order instead of an analysis of a demonstration. The extraction is called 'reasoning' hereinafter. Assumptions and an overview of the reasoning are presented in Section 2. In Section 3, a knowledge base for the reasoning is introduced. An algorithm of the reasoning is described in Section 4. In Section 5, the algorithm of the learning (i.e., updating the knowledge base) is introduced. In Section 6, both the credibility of the reasoning and the effectivity of the learning are evaluated; moreover, the feasibility of robot programming with reasoning is validated through programming of a door-opening task. The conclusions follow in Section 7.

2. OVERVIEW

2.1. Assumptions

The assumed service robot is a position-controlled mobile manipulator with a common hand (e.g., a two-finger hand). It performs object-handling tasks. Takase [5] reported that the tasks can be achieved by a sequence of motions. He classified the motions through glossaries of technical terms, as shown in Fig. 1. In this study, definitions are given which show that the robot can treat the tasks achieved by (2) and (3) because it is difficult for the robot to manipulate industrial tools or flexible objects. In addition, non-position-controlled motions in motions (1) are also out of scope.

However, the robot can treat objects that do not change their forms; it does not treat objects such as powder, liquid, living things, food and flexible objects. When these objects are in rigid containers, the robot can manipulate the containers instead of the objects.

2.2. Definition of task features

As mentioned in Section 1, teaching-by-showing has pre-defined task features. Since it is unrealistic to add new features at every new teaching, it is natural to consider applying features that are common in every task. The motions in Fig. 1 are potential features because tasks can be achieved by a sequence of motions. Among the motions, those in (1) are picked up since they can express the motions in (2), (3), (4) and (5). Motion (1) includes 18 motions, as shown in Fig. 2. Among the 18 motions, there are eight position-controlled motions; furthermore, the eight motions can be classified by kinds of motions on an object (the strict definition is 'motions on Grasp Point (GP) on an object'). Therefore, the above two motions — Linear

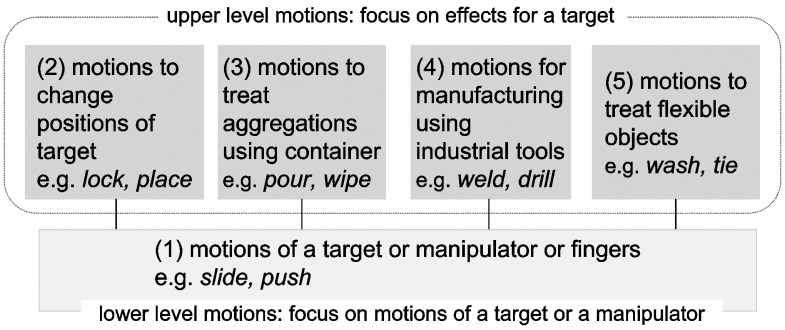

Figure 1. Classified motions. Motions from (2) to (5) can be expressed by combinations of motions in (1).

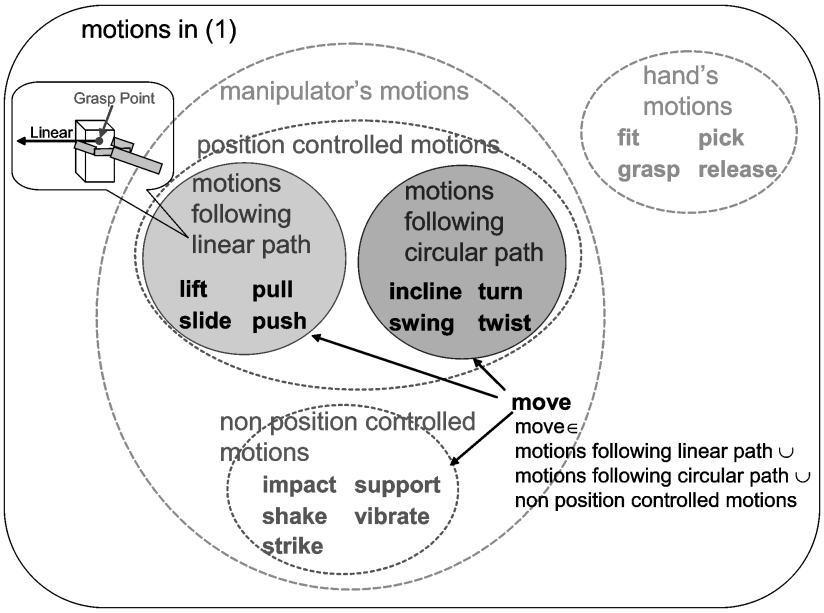

Figure 2. Classification of motions in motion (1).

motion and Circular motion — are adopted as the features. In addition, there are tasks that have only start and goal positions (e.g., fetch); the Point-To-Point (PTP) motion is added to the features. Almost all of the tasks that occur in a home or office can be expressed using combinations of Linear motion, Circular motion and PTP motion. For instance, when a task order 'turn off a printer' is executed, a switch of a printer moves with Linear motion. When 'open a refrigerator' is executed, a door on a refrigerator moves with Circular motion. When 'return a book' is executed, a book moves with PTP motion.

2.3. Format of reasoning

Reasoning means choosing motion(s) from a task order. Figure 3 is an overview of the reasoning. At input, the task order is formed with only two words, Task and Target, which form the most basic of task orders. Output is motion(s). An impossible task applies to a task that does not use manipulator or a dexterous task.

Any task order composed of natural language, which is rich (i.e., a large amount of words) and ambiguous (i.e., a word has multiple meanings), will complicate the reasoning. Expert systems [6–8] have primarily been used to deal with these complications. Furthermore, there are some studies that treat task understanding using semantics [9] or self-organizing memory [10]. However, their private manual knowledge bases have limitations with regard to the treatment of the richness and ambiguity of natural language. The problem has motivated the establishment of a Semantic Web [11]; unfortunately, it is still in its initial stages [12]. As a result, for the purposes of this study, an electronic thesaurus is used. Since the thesaurus is only a dictionary of related terms, knowledge for the reasoning is added. The knowledge refers to motions included in Fig. 1. The knowledge is propagated through hypernyms and hyponyms in the thesaurus, which enable reasoning from any task order. Moreover, the propagated knowledge has been updated (it is called 'learning' in this study) using the correct paths by a user in order to reinforce/customize the knowledge base. Details of the above algorithm will be described in the following sections.

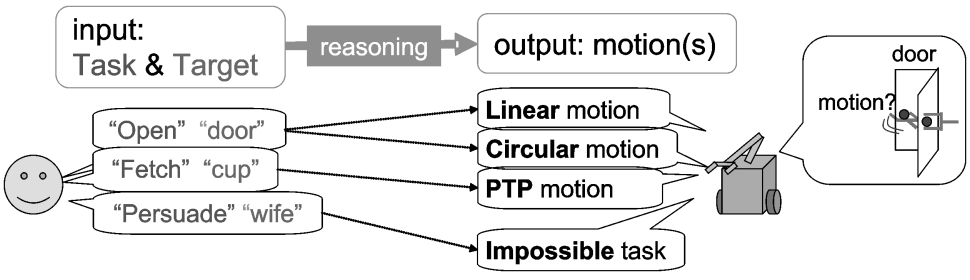

Figure 3. Input and output of reasoning.

3. BUILDING THE KNOWLEDGE BASE

3.1. Thesaurus — lexical database for the knowledge base

The applied thesaurus is WordNet2.0 [13]. Its design is inspired by current psycholinguistic and computational theories of human lexical memory. English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets. In this study, only verbs and nouns are applied since every Task/Target is a verb/noun.

WordNet has Hypernyms and Hyponyms when they are connected, they form trees (Fig. 4). The structure of the trees corresponds to the relationships between concepts of the words. Therefore, this study uses the trees as a lexical database for the knowledge base.

3.2. Knowledge for tasks and objects

The trees are just words classified through their similarities; the words are not associated with tasks/objects that the robot can manipulate. For instance, the reasoning system cannot understand the following sentences.

- *Turn* stands for a rotational motion.
- *Kindness* cannot be manipulated by a robot hand.

In this study, therefore, knowledge is added into the trees. The knowledge means words that clearly stand for a Linear motion, Circular motion, PTP motion and Impossible task.

The addition consists of two parts. The first step is an exception of words that do not express tasks/objects using lexicographer files included in WordNet. There are 45 lexicographer files, which form an index based on syntactic category and logical groupings. Table 1 shows part of the lexicographer files. The entire files are described in Ref. [14]. The following two production rules are definitions of the unperformable tasks or unmanageable targets based on Section 2.1.

(i) IF a Task is not included in verb.change, verb.contact or verb.motion, THEN a robot CANNOT execute the Task.

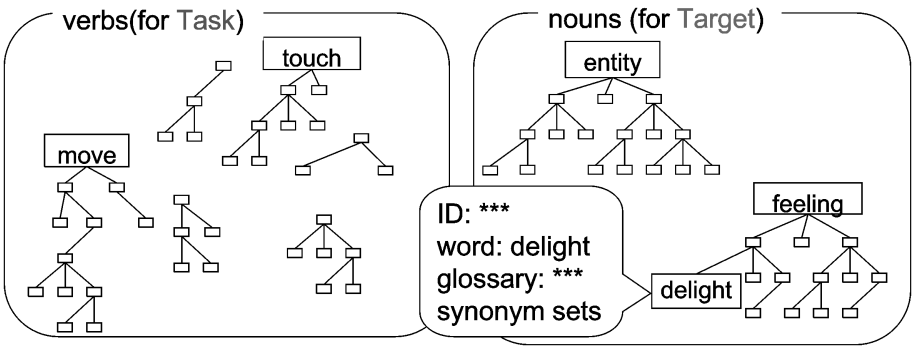

Figure 4. Trees of verbs and nouns.

(ii) IF a Target is not included in noun.artifact, THEN a robot CANNOT execute the Target.

As shown in Fig. 5, the production rules clip untreatable tasks and targets in the trees; if a user orders using the words included in the clipped parts, the system replies Impossible task.

The second step contains detailed definitions of words that stand for Linear motion, Circular motion, PTP motion or Impossible task included in verb.change, verb.contact, verb.motion and noun.artifact. Table 2 shows the defined rules. First, rules for Tasks (verbs) are defined. For the definitions of Linear motion and Circular motion (nos 1 and 2), the words in motion (1) in Fig. 2 are used. However, there is no word that stands for PTP motion in Fig. 2. The words from motion (2) are therefore selected (no. 3). Although the words representing untreatable tasks are derived

Table 1.
Lexicographer files

File number	Name	Contents
00	adj.all	all adjective clusters
05	noun.animal	nouns denoting animals
06	noun.artifact	nouns denoting man-made objects
08	noun.body	nouns denoting body parts
13	noun.food	nouns denoting foods and drinks
17	noun.object	nouns denoting natural objects e.g. mountain, island (not man-made)
18	noun.person	nouns denoting people
20	noun.plant	nouns denoting plants
27	noun.substance	nouns denoting substances
28	noun.time	nouns denoting time and temporal relations
29	verb.body	verbs of grooming, dressing, and bodily care
30	verb.change	verbs of size, temperature change, intensifying, and others
35	verb.contact	verbs of touching, hitting, tying, and digging
38	verb.motion	verbs of walking, flying, and swimming
44	adj.ppl	participial adjectives

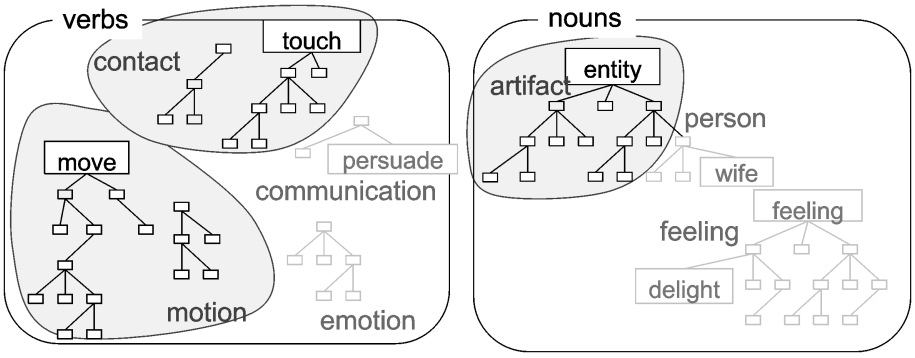

Figure 5. Clipping of untreatable tasks and targets.

from motions (1), (4) and (5), the words themselves in (4) and (5) are not used since some of the words do not strictly satisfy the definitions of the motions (e.g., although motion (5) contains *wash*, *wash* can be used for both flexible and rigid objects). Moreover, it is not reasonable to define Impossible task for all motions in (4) and (5), although reasoning with minimum knowledge is this paper's policy. Therefore, instead of words, a more conceptual definition is used for (4) and (5) in no. 5.

Next, rules for Targets (nouns) are defined. Since nouns have no definitions of the motions such as Fig. 2, nouns (Targets) that are usually moved with Linear motion, Circular motion or PTP motion during their manipulations are defined as knowledge. Concretely, nouns whose glossaries contain words that stand for Linear motion, Circular motion or PTP motion (i.e. words in nos 1–3) are selected in nos 6–8. In addition, nouns (Targets) that have the parts which cause the Targets to be moved with Linear motion or Circular motion are selected in nos 10 and 11. However, in no. 9, the words that stand for an untreatable target are derived from Section 2.1.

3.3. Addition of probabilities and certainties

To propagate the knowledge in Table 2 throughout the trees and then enable the reasoning, this study defines a set of

- $A_m(w)$: an accordance between word (w) and motion (m), and
- $C_m(w)$: certainty factor for $A_m(w)$ for each word,

as shown in Fig. 6. In the case of w = verb (i.e., w = Task), for instance, $A_{\mathrm{ccr}}(rotate) = 1.0$ means that 'an object moves with Circular motion when it is *rotated*'. In the case of w = noun (i.e., w = Target), for instance, $A_{\mathrm{lnr}}(sliding\ door) = 1.0$ means that 'there is a *sliding door* that moves with Linear motion'. Moreover, $C_{\mathrm{lnr}}(sliding\ door) = 0.5$ means that the system has moderate (50%) certainty for $A_{\mathrm{lnr}}(sliding\ door) = 1.0$.

Four sets of $A_m(w)$ and $C_m(w)$ are set in each word: $A_{\mathrm{lnr}}(w)$ and $C_{\mathrm{lnr}}(w)$, $A_{\mathrm{ccr}}(w)$ and $C_{\mathrm{ccr}}(w)$, $A_{\mathrm{ptp}}(w)$ and $C_{\mathrm{ptp}}(w)$, $A_{\mathrm{imp}}(w)$ and $C_{\mathrm{imp}}(w)$. Note that each $A_m(w)$ is

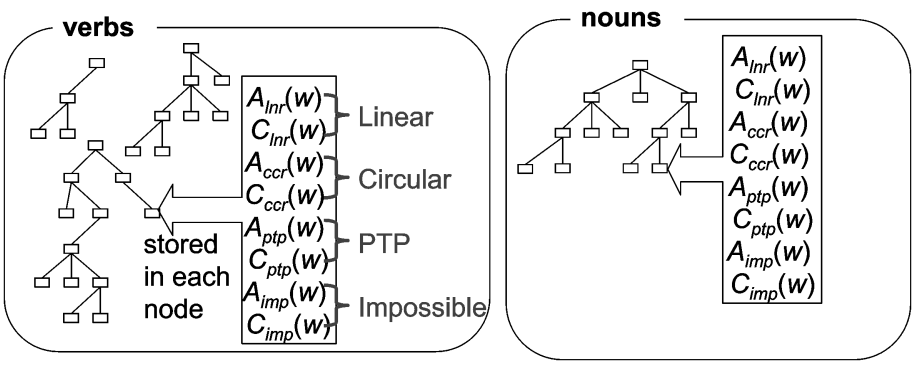

Figure 6. Setting of $A_m(w)$ and $C_m(w)$.

independent; the independence can express compatibility with words. (In this study, 'compatibility' means 'a word includes two contradictory meanings'. 'Ambiguity' means 'one word can be interpreted in various meanings'.) For instance, '*open* stands for Linear or Circular motion depending on Target' can be expressed using $A_{\mathrm{lnr}}(open) = 1.0$ and $A_{\mathrm{ccr}}(open) = 1.0$.

3.4. Setting values of probabilities and certainties

The values on $A_m(w)$ and $C_m(w)$ in each node in the trees are set using the production rules in Table 2. First, the rules are converted to values in $A_m(w)$ and $C_m(w)$. Table 3 shows parts of the values.

Table 2.
Production rules

For Task (verb.change, verb.contact, verb.motion)	
Identification of words themselves	
No. 1	IF Task is *lift*, *pull*, *push* or *slide*, THEN a object moves with a Linear motion, NOT moves with a Circular nor PTP motion and a robot can execute the Task.
No. 2	IF Task is *incline*, *swing*, *turn* or *twist*, THEN a object moves with a Circular motion, NOT moves with a Linear nor PTP motion and a robot can execute the Task.
No. 3	IF Task is *bring*, *transfer* or *carry*, THEN a object moves with a PTP motion, NOT moves with a Linear nor Circular motion and a robot can execute the Task.
No. 4	IF Task is *impact*, *shake*, *strike*, *support* or *vibrate*, THEN a robot CANNOT execute the Task.
Identification of words in glossaries	
No. 5	IF Task has *with tool*[a] or *flwxible form* in its glossary THEN a robot CANNOT execute the Task.
Identification of words in holonyms	
	N/A
For Target (noun.artifact)	
Identification of words themselves	
	N/A
Identification of words in glossaries	
No. 6	IF Target has *slide*, *pull* or *push* in its glossary, THEN Target moves with a Linear motion and a robot can manipulate the Target.
No. 7	IF Target has *turn*, *rotate* or *swing* in its glossary, THEN Target moves with a Circular motion and a robot can manipulate the Target.
No. 8	IF Target has *bring*, *transfer* or *carry* in its glossary, THEN Target moves with a PTP motion and a robot can manipulate the Target.
No. 9	IF Target has *flexible*, *powder* or *liquid* in its glossary, THEN a robot CANNOT execute the Task.
Identification of words in holonyms	
No. 10	IF Target has *linear guide* as its part, THEN Target moves with a Linear motion and a robot can manipulate the Target.
No. 11	IF Target has *hinge* as its part, THEN Target moves with a Circular motion and a robot can manipulate the Target.

[a] WordNet uses *tool* as an *industrial tool*.

Table 3.
Initial probabilities and certainties

No.	$A_m(w)$	$C_m(w)$
1	$A_{\text{lnr}}(slide) = 1.0$ $A_{\text{ccr}}(slide) = 0.0$ $A_{\text{ptp}}(slide) = 0.0$ $A_{\text{imp}}(slide) = 0.0\ldots$ $A_{\text{lnr}}(pull) = 1.0\ldots$ (total 16 pieces)	$C_{\text{lnr}}(slide) = 1.0$ $C_{\text{ccr}}(slide) = 1.0$ $C_{\text{ptp}}(slide) = 1.0$ $C_{\text{imp}}(slide) = 1.0\ldots$ $C_{\text{lnr}}(pull) = 1.0\ldots$ (total 16 pieces)
5	$A_{\text{imp}}(bore) = 1.0$ (total 6 pieces)	$C_{\text{imp}}(bore) = 0.8$ (total 6 pieces)
7	$A_{\text{ccr}}(screw) = 1.0\ldots$ $A_{\text{ccr}}(winch) = 1.0\ldots$ (total 460 pieces)	$C_{\text{ccr}}(screw) = 0.8\ldots$ $C_{\text{ccr}}(winch) = 0.8\ldots$ (total 460 pieces)
10	N/A	N/A
11	$A_{\text{ccr}}(gate) = 1.0$ $A_{\text{ccr}}(swing\_door) = 1.0\ldots$ (total 12 pieces)	$C_{\text{ccr}}(gate) = 1.0\ldots$ $C_{\text{ccr}}(swing\_door) = 1.0\ldots$ (total 12 pieces)

Many words in Table 2 have multiple meanings; hence, authors determined whether or not each meaning matched the rule by using its glossary. Nevertheless, authors were not able to discriminate the multiple meanings from rules nos 5–9, which indicate associated words in the glossary with specially developed software. Therefore, this study sets $C_m(w) = 0.8$ for the penalty.

Next, the sets of $A_m(w)$ and $C_m(w)$ in Table 3 are stored in the tree; then, the sets are propagated to other sets without values, as shown in Fig. 7. This process is performed for each motion (m = lnr, ccr, ptp, imp) independently. The system starts a depth-first search from the root node. If it reaches a node with a value, it calculates $A_m(w)$ and $C_m(w)$ in its adjoining nodes that have no value. The calculation method depends on the hierarchical relationship between the current node and the adjoining nodes. The following is the basic principle to define the method:

- A child node inherits concepts of a parent node.
- Concepts of a parent are weighted averages of concepts of child nodes.
- Among brother nodes, only concepts that are inherited by a parent node are common. It is unknown whether unique concepts of brother nodes are common.

Furthermore, in order to adopt the value with the most reliable concept, the node(s) within only one hierarchy is used for the calculation even if nodes in different hierarchies have values. The priority of the hierarchy is the order of parents, a child and a brother. Equations for the calculation are shown in (1)–(6) (refer to Fig. 7). Using parent node:

$$A_m(w) = A_m(p), \tag{1}$$

$$C_m(w) = C_m(p), \tag{2}$$

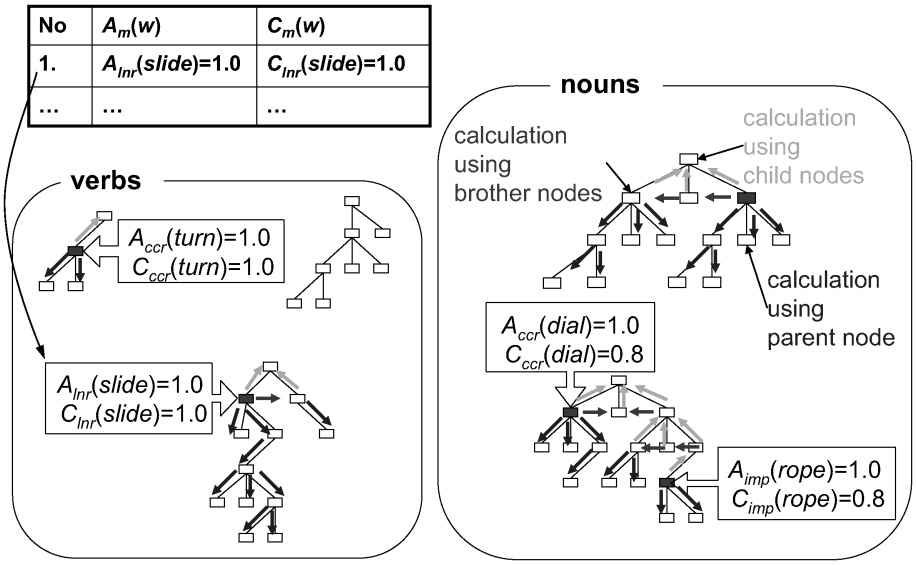

Figure 7. Propagation of $A_m(w)$ and $C_m(w)$.

p represents the parent node of w.

Using child nodes:

$$A_m(w) = \frac{\sum_{i=1}^{n}(A_m(c_i) \cdot C_m(c_i))}{\sum_{i=1}^{n} C_m(c_i)}, \tag{3}$$

$$C_m(w) = \frac{\sum_{i=1}^{n} C_m(c_i)}{n}, \tag{4}$$

c represents the child node of w, i is the number of child nodes and n is quantity of child nodes.

Using brother nodes:

$$A_m(w) = \frac{\sum_{i=1}^{n}(A_m(b_i) \cdot C_m(b_i))}{\sum_{i=1}^{n} C_m(b_i)}, \tag{5}$$

$$C_m(w) = 0.0, \tag{6}$$

b represents the brother node of w, i is the number of brother nodes and n is quantity of brother nodes.

To use $A_m(w)$ with high certainty, (3) and (5) apply the weighted average with $C_m(w)$ as their weightings. Equations (3) and (4) are executed after all child nodes are searched. Equations (5) and (6) are also executed after all brother nodes are searched; moreover, only sets of $A_m(b_i)$ and $C_m(b_i)$ with values are used for the calculation. If brother nodes are used in the calculation, their parent node has no value due to the priority. Namely, the values in $A_m(b_i)$ and $C_m(b_i)$ are not inherited by the parent. Therefore, $C_m(w)$ in (6) is 0.0 following the last item in the

above principle (i.e., it is unknown whether unique concepts of brother nodes are common).

After a value in an adjoining node is calculated, the value is also used to calculate its adjoining nodes without any value. Therefore, the calculations propagate the values in Table 3. To utilize not only the inheritance from parent nodes, but also tracking back the hierarchy using values of child and brother nodes, enables the system to set values in all nodes, even if a small number of nodes receive the initial values. However, the system cannot propagate the values when no node receives the value in Table 3, such as a center tree in Fig. 7.

4. REASONING

After setting the values in $A_m(w)$ and $C_m(w)$, the reasoning is ready. The process of the reasoning is as follows:

(i) Judgment whether task order is executable or not.

(ii) Decision of a priority about Task or Target.

(iii) Reasoning of motion(s) (Fig. 8).

(iv) Removal of discrepancies of the motion(s) between Task and Target.

In (i), the system judges an impossible task order, as in Fig. 5. This process is a screening before real reasoning; an impossible task order is reasoned using $A_{\mathrm{imp}}(w)$ and $C_{\mathrm{imp}}(w)$ again.

In (ii), the system uses Task preferentially for the reasoning because Task generally assumes responsibility for the object's motion. It uses Target only when $A_m(Task)$ and $C_m(Task)$ of all motions are without believability. In this study, 'without believability' is defined as $A_m(w) \cdot C_m(w) < motion\_thr$, and the value of $motion\_thr$ is 0.25 in this study. 0.25 was defined by intuition of the authers. In addition, if $A_m(w) \cdot C_m(w) < motion\_thr$ in both Task and Target, the system also applies Task.

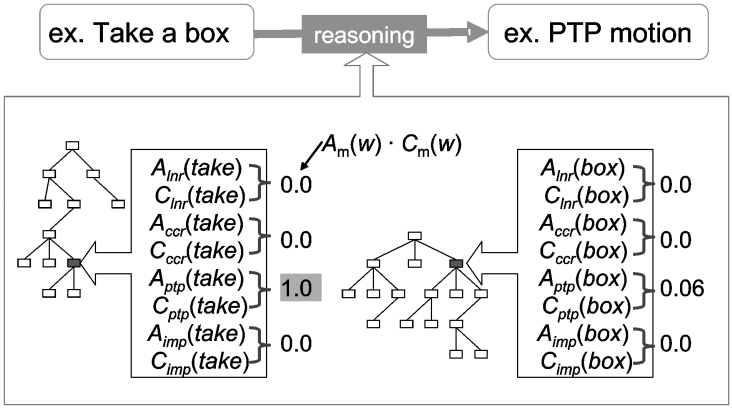

Figure 8. Reasoning of motion.

In (iii), the system examines m_ans that means motion(s), which have a maximum $A_m(w) \cdot C_m(w)$ and its around values, as shown in:

$$m\_ans = \{m^*, m^{**}\}, \tag{7}$$

where:

$$m^* = \underset{m}{\text{argmax}}\{A_m(w) \cdot C_m(w)\},$$

$$m^{**} = \begin{cases} \{m | A_m(w) \cdot C_m(w) \geqslant mlt\} & A_{m^{*\prime}}(w) \geqslant 0 \cap C_{m^{*\prime}}(w) \geqslant 0 \\ \{m | A_m(w) + C_m(w) \geqslant add\} & \text{else if } add \geqslant 0 \\ \{A_m(w) + C_m(w) \geqslant 0\} & \text{otherwise,} \end{cases}$$

$$mlt = A_{m^{*\prime}}(w) \cdot C_{m^{*\prime}}(w),$$

$$add = A_{m^{*\prime}}(w) + C_{m^{*\prime}}(w),$$

$$A_{m^{*\prime}}(w) = A_{m^*}(w) - A_{thr},$$

$$C_{m^{*\prime}}(w) = C_{m^*}(w) - C_{thr}.$$

If all motions $A_m(w) \cdot C_m(w)$ are 0.0, the system uses only $A_m(w)$ for the reasoning. In addition, in this study, the values of both A_{thr} and C_{thr} are fixed as 0.2, as defined by intuition of the authors.

In (IV), if the system uses Task for the reasoning and the number of reasoned motions is larger than one, there remains a motion(s) that satisfies $A_{m\_ans}(Target) \cdot C_{m\_ans}(Target) \geqslant motion\_thr$ because $A_m(Target) \cdot C_m(Target)$ has not been checked if Task is used for the reasoning except $A_m(Task) \cdot C_m(Task) < motion\_thr$. If no motion remains, the above canceling is invalid and all canceled motions are backed in the solution set. However, if the system uses Target, it cancels reasoned motion(s) with $A_{m\_ans}(Task) = 0.0$ and $C_{m\_ans}(Task) = 1.0$ because $A_m(Task) \cdot C_m(Task)$ has already been checked.

5. LEARNING

The knowledge base has not filled all values of $A_m(w)$ and $C_m(w)$ such as the center upper tree in Fig. 7. Moreover, there are no correct answers because unique common knowledge cannot be defined. For instance, the form of a faucet differs according to the environment in which a robot works; faucets with rotating handle are common in some homes and faucets with push-buttons or levers are common in other homes. Therefore, a faucet's motion to turn on/off should be changed according to the faucet's form. To reinforce and customize the knowledge base, the system updates $A_m(w)$ and $C_m(w)$ using input from a user. The term 'learning' refers to this updating.

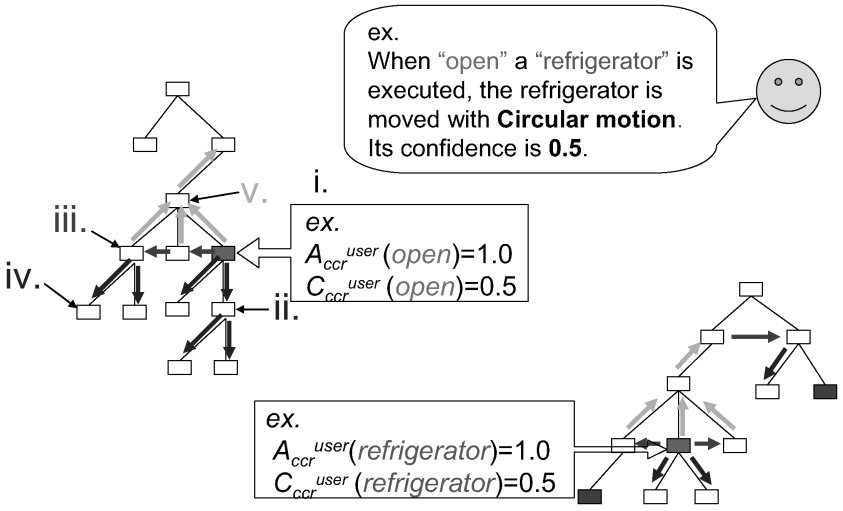

Figure 9. Update process of $A_m(w)$ and $C_m(w)$.

First, a user gives new accordance $A_m^{\text{user}}(w)$ and its certainty $C_m^{\text{user}}(w)$ in both Task and Target. While the value of all $A_m^{\text{user}}(w)$ is fixed as 1.0, the value of $C_m^{\text{user}}(w)$ can be changed by the user. After the input, the system updates $A_m(w)$ and $C_m(w)$ and their adjoining nodes $A_m(w)$ and $C_m(w)$. The update progresses under the following steps (refer to Fig. 9):

(i) Current node.

(ii) All lower nodes of (i).

(iii) Brother nodes of (i).

(iv) All lower nodes of (iii).

(v) Parent node of (i).

During the update, it also considers the hierarchical relationship in Section 3.4.

In (i), the system calculates $A_m^{\text{tmp}}(w)$ and $C_m^{\text{tmp}}(w)$: temporary updating values, as shown in:

$$A_m^{\text{tmp}}(w) = A_m^{\text{user}}(w), \tag{8}$$

$$C_m^{\text{tmp}}(w) = C_m^{\text{user}}(w) \cdot \alpha. \tag{9}$$

α is a certainty factor. It is applied to prevent modification of knowledge defined in Table 3 by uncertain $A_m^{\text{user}}(w)$ and $C_m^{\text{user}}(w)$. Its value is 0.5 in this study, as defined by intuition of the authors.

Next, the system calculates $A_m^{j+1}(w)$ and $C_m^{j+1}(w)$: fixed updating values. The equations are changed according to a closeness between $C_m^{\text{tmp}}(w)$ and $C_m^{j}(w)$, as

shown in:

$$A_m^{j+1}(w) = \begin{cases} A_m^{\mathrm{tmp}}(w) & C_m^{\mathrm{tmp}}(w) > C_m^j(w) + C\_thr \\ A_m^j(w) & C_m^{\mathrm{tmp}}(w) < C_m^j(w) - C\_thr \\ \dfrac{A_m^{\mathrm{tmp}}(w) \cdot C_m^{\mathrm{tmp}}(w) + A_m^j(w) \cdot C_m^j(w)}{C_m^{\mathrm{tmp}}(w) + C_m^j(w)} & \text{otherwise,} \end{cases} \tag{10}$$

$$C_m^{j+1}(w) = \begin{cases} C_m^{\mathrm{tmp}}(w) & C_m^{\mathrm{tmp}}(w) > C_m^j(w) + C\_thr \\ C_m^j(w) & C_m^{\mathrm{tmp}}(w) < C_m^j(w) - C\_thr \\ \dfrac{C_m^{\mathrm{tmp}}(w) + C_m^j(w)}{2} & \text{otherwise.} \end{cases} \tag{11}$$

Suffix j means before update and $j+1$ means after update. C_thr is already defined in Section 4. Furthermore, if $C_m^{\mathrm{tmp}}(w) < C_m^j(w) - C\_thr$, the system quits to go to the next step.

In (ii) and (iv), the system calculates $A_m^{\mathrm{tmp}}(w)$ and $C_m^{\mathrm{tmp}}(w)$, as shown in:

$$A_m^{\mathrm{tmp}}(w) = A_m(p), \tag{12}$$

$$C_m^{\mathrm{tmp}}(w) = C_m(p), \tag{13}$$

p represents the parent node of w.

Next, the system calculates $A_m^{j+1}(w)$ and $C_m^{j+1}(w)$ using (10) and (11).

In (iii), the system calculates $A_m^{\mathrm{tmp}}(w)$ and $C_m^{\mathrm{tmp}}(w)$, as shown in:

$$A_m^{\mathrm{tmp}}(w) = \frac{\sum_{i=1}^{n}(A_m(b_i) \cdot C_m(b_i))}{\sum_{i=1}^{n} C_m(b_i)}, \tag{14}$$

$$C_m^{\mathrm{tmp}}(w) = 0.0. \tag{15}$$

b represents the brother node of w, i is the number of brother nodes and n is a quantity of brother nodes, which includes nodes of w and step (i).

Next, the system calculates $A_m^{j+1}(w)$ and $C_m^{j+1}(w)$ using (10) and (11).

In (v), the system calculates $A_m^{\mathrm{tmp}}(w)$ and $C_m^{\mathrm{tmp}}(w)$, as shown in:

$$A_m^{\mathrm{tmp}}(w) = \frac{\sum_{i=1}^{n}(A_m(c_i) \cdot C_m(c_i))}{\sum_{i=1}^{n} C_m(c_i)}, \tag{16}$$

$$C_m^{\mathrm{tmp}}(w) = \frac{\sum_{i=1}^{n} C_m(c_i)}{n}, \tag{17}$$

c represents the child node of w, i is the number of child nodes and n is a quantity of child nodes. Next, the system calculates $A_m^{j+1}(w)$ and $C_m^{j+1}(w)$ using (10) and (11).

After (v) has been performed, the system resets the parent node to the current node and repeats from (iii) to (v). This causes the user's input to be propagated through the tree.

6. REASONING EXPERIMENTS

This chapter introduces experiments in which the system determines motions on the basis of various task orders. First, the results of reasoning using the knowledge base limited to the initial knowledge in Section 3.2 are introduced. Next, reasoning results after learning are introduced.

6.1. Results using only initial knowledge

First, the reasoning system screened the input Task and Target to find any Impossible tasks orders using lexicographer files (Section 3.2). Table 4 shows some results of Impossible tasks.

Next, the system reasoned a motion(s) using $A_m(w)$ and $C_m(w)$. Figure 10 shows the results of the reasoning and their $A_m(w) \cdot C_m(w)$ from six task orders. In the graph, $A_m(w) \cdot C_m(w)$ corresponding to answers show high values. At *cut rope*, the answer was Impossible task because *line*, which is a parent node of *rope*, corresponds to rule no. 9 in Table 2. At *take box*, the answer was PTP motion since *take* and *bring* have the same meaning (i.e., they are stored in the same node) on WordNet; therefore, *take* corresponds to rule no. 3. At *lock key*, the answer was Circular motion because *key* corresponds to rule no. 7. No values within the graph such as *lock* mean null, which was not calculated due to lack of the corresponding production rules, as mentioned in Section 3.4. However, at *fetch key*, the answer was PTP motion because the value of $A_{\mathrm{ptp}}(fetch) \cdot C_{\mathrm{ptp}}(fetch)$ was larger than the value of $A_{\mathrm{ccr}}(key) \cdot C_{\mathrm{ccr}}(key)$. At *open door* and *open drawer*, although all motions $A_m(open) \cdot C_m(open)$ was null, the system selected correct answers using $A_m(drawer) \cdot C_m(drawer)$ or $A_m(door) \cdot C_m(door)$. *drawer* corresponds to rule no. 6, and *door* corresponds to rules nos 6 and 7. Note that the reasoned motions are not always specified by the compatibility of words such as *door*.

6.2. Results after learning

The percentage of right answers versus the number of leaning opportunities was evaluated. The following list steps of the learing experiment. First, the reasoning system reasoned abstract motions for the 10 test data shown in Table 5. Second, a user fed training data shown in Table 6 into the reasoning system. Third,

Table 4.
Reason of impossible tasks

Task	Target	Cause of Impossible task
connect	*Internet*	*connect* belongs to lexi. file ‘communication’.
input	*data*	*data* belong to lexi. file ‘group’.
persuade	*wife*	*wife* belongs to lexi. file ‘person’.
pour	*coffee*	*coffee* belongs to lexi. file ‘food, drink’.
sew	*skirt*	*sew* belongs to lexi. file ‘creation’.

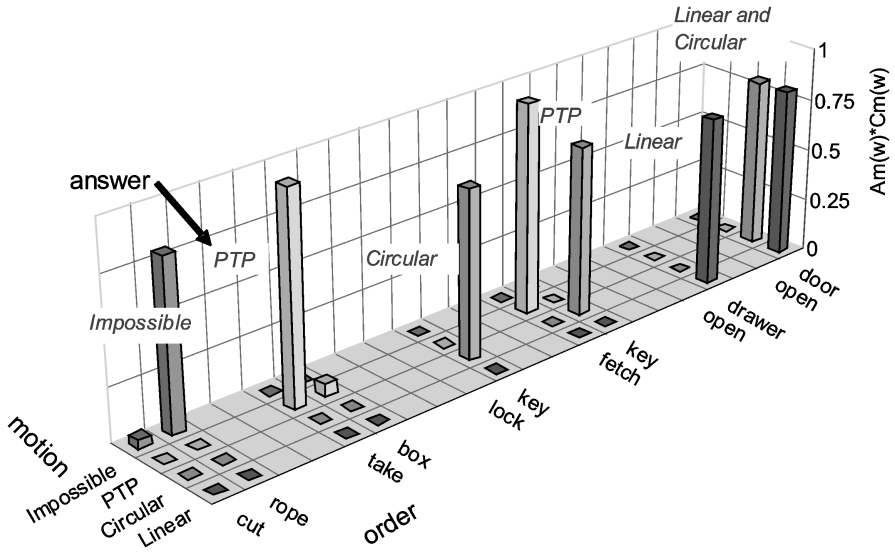

Figure 10. Reasoning results through only initial knowledge.

Table 5.
Results of learning

Task	Target	Answer	Reasoning after learning opportunities				
			0[a]	10	20	30	40
close	door	**L or C**[b]	**L or C**	C	C	C	**L or C**
close	drawer	**L**	**L**	C	C	C	**L**
insert	plug	**L**	C	C	C	C	C
lift[c]	box	**P**	L or C	L or C	L or C	L or C	L or C
mop	corridor	**L**	A	A	A	**L**	**L**
open	refrigerator	**C**	P	P	P	P	P
return	book	**P**	**P**	**P**	**P**	**P**	**P**
turn off	printer	**L**	L or C	**L**	**L**	**L**	**L**
take away	glass	**P**	**P**	**P**	**P**	**P**	**P**
vacuum	floor	**L**	C	C	**L**	**L**	**L**
Right answers ratio		1.0	0.4	0.3	0.4	0.5	0.7

[a] The number of learning opportunities.
[b] L: Linear, C: Circular, P: PTP, I: Impossible, A: All kinds of motions.
[c] *Lift* in this experiment means 'take hold of something and move it to a different location', not 'raise from a lower to a higher position'.

the reasoning system propagated the training data. Finally, the reasoning system reasoned abstract motions for the test data again. The second step to the last step were repeated until the 40th learing stage.

Table 5 shows the results of learning about typical domestic tasks. Table 6 shows trained data that caused changes in the results. The ratio of right answers was 0.4 at

Table 6.
Beneficial training data

No.	Task	Target	training data	effect on test data
1	close	cap	$A^{\text{user}}_{\text{ccr}}(close) = 1.00$ $C^{\text{user}}_{\text{ccr}}(close) = 1.00$ $A^{\text{user}}_{\text{ccr}}(cap) = 1.00$ $C^{\text{user}}_{\text{ccr}}(cap) = 1.00$	$A_{\text{ccr}}(close) = \text{null} \rightarrow 1.00$ $C_{\text{ccr}}(close) = \text{null} \rightarrow 0.50$
6	switch off	stereo	$A^{\text{user}}_{\text{lnr}}(switchoff) = 1.00$ $C^{\text{user}}_{\text{lnr}}(switchoff) = 1.00$ $A^{\text{user}}_{\text{lnr}}(stereo) = 1.00$ $C^{\text{user}}_{\text{lnr}}(stereo) = 1.00$	$A_{\text{lnr}}(turn\ off) = 0.36 \rightarrow 1.00$ $C_{\text{lnr}}(turn\ off) = 0.00 \rightarrow 0.50$
18	sweep	entrance	$A^{\text{user}}_{\text{lnr}}(sweep) = 1.00$ $C^{\text{user}}_{\text{lnr}}(sweep) = 1.00$ $A^{\text{user}}_{\text{lnr}}(entrance) = 1.00$ $C^{\text{user}}_{\text{lnr}}(entrance) = 1.00$	$A_{\text{lnr}}(vacuum) = \text{null} \rightarrow 1.00$ $C_{\text{lnr}}(vacuum) = \text{null} \rightarrow 0.00$
25	swab	desk	$A^{\text{user}}_{\text{lnr}}(swab) = 1.00$ $C^{\text{user}}_{\text{lnr}}(swab) = 1.00$ $A^{\text{user}}_{\text{lnr}}(desk) = 1.00$ $C^{\text{user}}_{\text{lnr}}(desk) = 1.00$	$A_{\text{lnr}}(mop) = \text{null} \rightarrow 1.00$ $C_{\text{lnr}}(mop) = \text{null} \rightarrow 0.16$
31	close	shutter	$A^{\text{user}}_{\text{lnr}}(close) = 1.00$ $C^{\text{user}}_{\text{lnr}}(close) = 1.00$ $A^{\text{user}}_{\text{lnr}}(shutter) = 1.00$ $C^{\text{user}}_{\text{lnr}}(shutter) = 1.00$	$A_{\text{lnr}}(close) = 1.00 \rightarrow 1.00$ $C_{\text{lnr}}(close) = 0.11 \rightarrow 0.50$

time 0 and after it dropped to 0.3, it rose to 0.7 at time 40. That is, the answers of the reasoning move gradually closer to the correct answers through learning.

At the learning of words including compatibility, oscillation of their reasoning results is a significant point. For instance, motions of *close* change to Linear motion or Circular motion depending on objects. When $A_m(close)$ and $C_m(close)$ are updated, there is a possibility that reasoning results oscillate between Linear motion and Circular motion. At *close door* and *close drawer*, the reasoning results of the proposed method, which has independent $A_m(w)$, $C_m(w)$ in each motion, oscillates until no. 30 when unfilled $A_m(w)$, $C_m(w)$ exests. However, the results converge on correct answers from no. 31 when all values of $A_m(w)$, $C_m(w)$ are filled. Therefore, the proposed method can reason/learn in consideration of the compatibility.

6.3. Application

As an application of the reasoning, a system that navigates robot programming by an amateur user was introduced. The reasoned motion was used to make directions for the user. Figure 11 shows pictures of robot programming for a door-opening task. The following description is its digest.

(i) A user inputs *open* and *door* to the reasoning system as a task order (Fig. 11a).

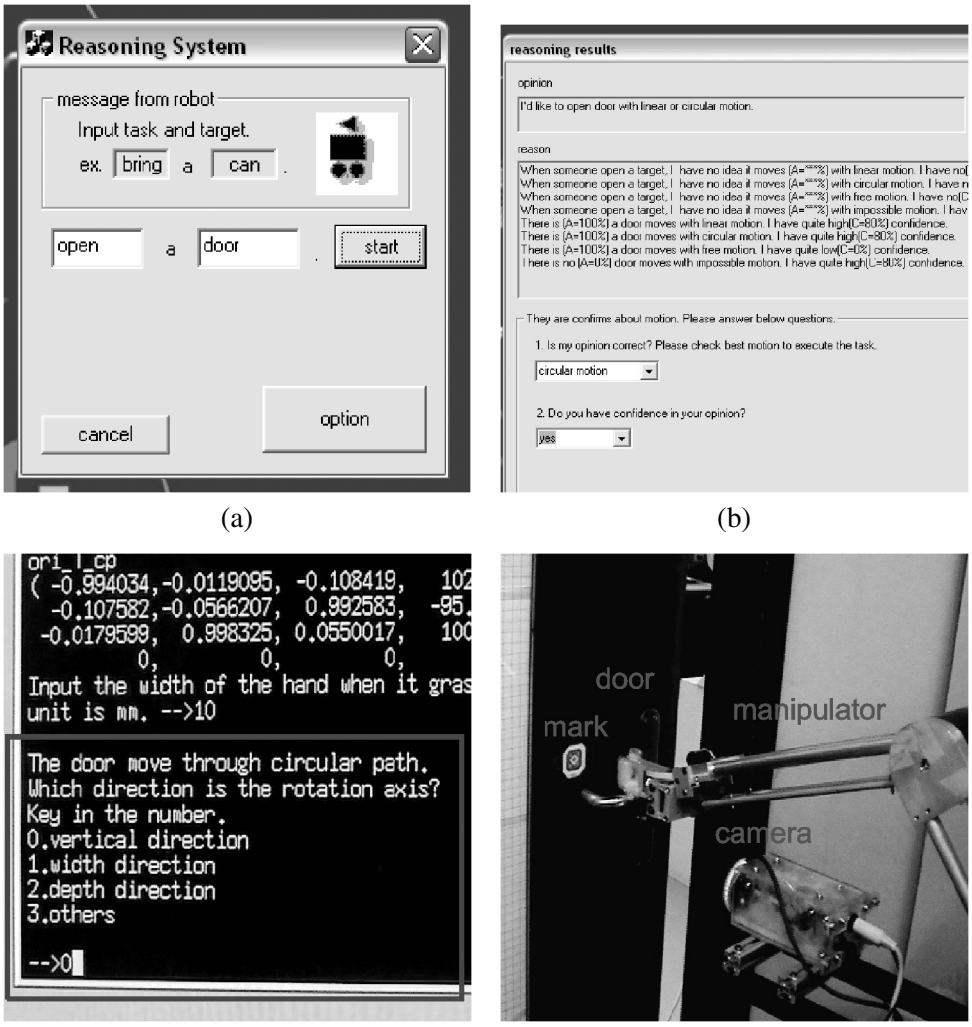

Figure 11. Robot programming of a door-opening task: (a) input *open* and *door*; (b) reasoning result; (c) input rotation axis; (d) playback.

(ii) As shown in Fig. 11b, the system reasoned the motion. In 'opinion' on the dialog box, the system output the reasoned motions. $A_m(w)$ and $C_m(w)$ in *open* and *door* can be seen in 'reason' on the dialog box. The system displayed Linear or Circular motion as the answer because the concept of *door* includes two types of door: a swing door and a sliding door. In this situation, the user should identify the motion; the user input 'Circular motion' in Q.1 below the dialog.

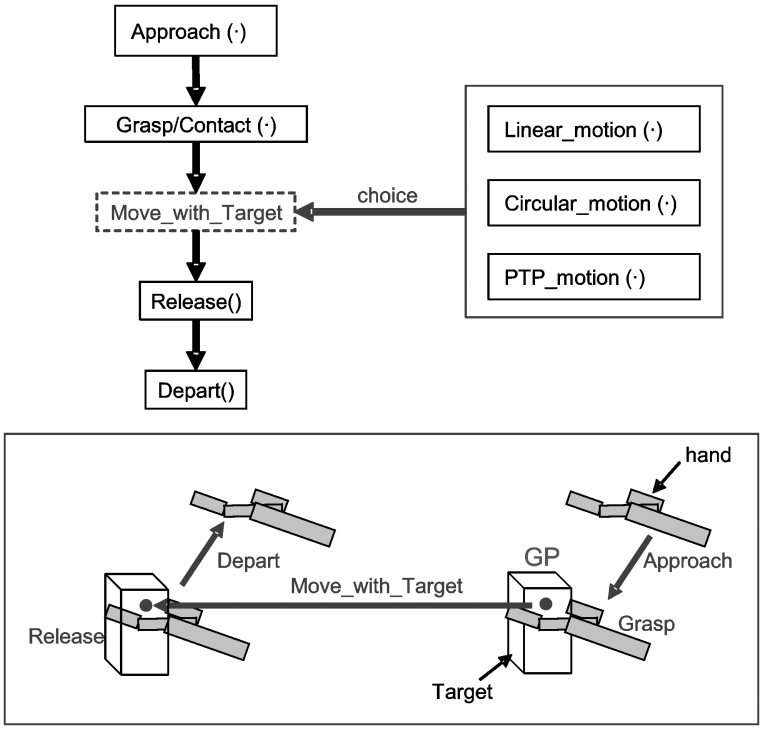

Figure 12. Action sequence. Arguments in each action are used to create an actual motion.

(iii) As shown in Fig. 12, the system already has a sequence of the robot's actions to realize the reasoned motion. The system stored Circular_motion(·) in Move_with_target.

(iv) The system asked the user to fill up the arguments in each action. Figure 11c is a snapshot of the inquiry for the rotational axis.

(v) The robot executed *open* the *door* using the sequence (Fig. 11d). In addition, the robot measured the pose of the doorknob using the pose of the mark. The pose of the mark was measured through a camera image.

The system including the reasoning algorithm enabled us to figure out the abstract motion of the object before it was programmed as the robot's action. Thus, the system can also request a demonstration of a specific motion to acquire certain data (e.g., 'Please grasp the door'). Moreover, the untrained user can program tasks because the system informs the user of what the system wants to be taught. In addition, if the reasoning algorithm is mounted in a teaching-by-showing system, the system can remove sensory data error using information of a reasoned abstract motion of an object. Hammel *et al.* [15] have developed DeVAR (Desktop Vocational Assistant Robot) and have enumerated 46 robotic tasks for individuals with high-level quadriplegia. The 46 tasks include meal preparation/feeding, vocational, hygiene, recreational and miscellaneous tasks. To

enumerate tasks handled by a service robot, we extracted 33 robotic tasks belonging to meal preparation tasks and vocational tasks from the 46 tasks. Then, we checked whether the 33 tasks can be executed with one motion: Linear motion, Circular motion and PTP motion. As a result, 55% of the checked tasks can be executed with one motion. Therefore, the reasoning is useful for the teaching of service robots.

7. CONCLUSIONS

For the pre-knowledge of object-handling-task programming, in this paper an attempt is made to reason an abstract motion of a target object through a task order, which consists of two words, Task and Target (e.g., 'switch on' and 'light'). The kinds of motions are defined as Linear, Circular and Point-to-Point.

The core of the reasoning is the utilization of closeness among the meaning of words. Moreover, a thesaurus makes it possible to use rich and ambiguous natural language. Learning, which is the on-site updating of the knowledge base by the user, achieved reinforcement/customization of the knowledge base. The effectiveness of the proposed method was proved through three experiments: the reasoning, learning and programming of a door-opening task.

This reasoning method can be used for not only the reasoning of abstract motion, but reasoning of abstract grasping force, etc. (e.g., abstract grasp force can be reasoned using this knowledge: 'IF glossary of the Target contains *fragile*, a robot has to grasp Target delicately'). Moreover, the method can also be expanded for other applications (e.g., dialogue understanding for communication robots). One problem is that the method cannot reason tasks that consist of more than one motion (e.g., when we open a plastic bottle, we turn and lift its lid). It is assumed that the questions from the reasoning system to the user represent an efficient solution. For instance, the system asks whether or not another motion(s) is needed when a task's playback fails.

Interest in a Semantic Web has been increasing. When the knowledge about the property of objects and a robot's motions to realize various tasks are uploaded and the reasoning system can use it, the system will be able to reason not only objects motion but also object manipulation strategy. Its reasoning style will be more natural and resemble that of humans.

REFERENCES

1. Y. Kuniyoshi, M. Inaba and H. Inoue, Learning by watching: extracting reusable task knowledge from visual observation of human performance, *IEEE Trans. Robotics Automat.* **10**, 799–822 (1994).
2. Y. Maeda, N. Ishido, H. Kikuchi and T. Arai, Teaching of grasp/graspless manipulation for industrial robots by human demonstration, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne, pp. 1523–1528 (2002).

3. J. Takamatsu, H. Tominaga, K. Ogawara, H. Kimura and K. Ikeuchi, Extracting manipulation skills from observation, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Takamatsu, pp. 584–589 (2000).
4. A. Billard, Y. Epars, G. Cheng and S. Schaal, Discovering imitation strategies through categorization of multi-dimensional data, in: *Proc. 2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Las Vegas, NV, pp. 2398–2403 (2003).
5. K. Takase, Study on design and control of torque-controlled manipulators, *Res. Electrotechnical Lab.* **876** (1986) (in Japanese).
6. B. G. Buchanan and E. A. Feigenbaum, DENDRAL and meta-DENDRAL: their applications dimension, *Artificial Intell.* **11**, 5–24 (1978).
7. H. E. Pople, Jr., J. D. Myers and R. A. Miller, DIAROG: a model of diagnostic logic for internal medicine, in: *Proc. of Int. Joint. Conf. Artificial Intellifence*, Tbilisi, pp. 848–855 (1975).
8. E. H. Shortliffe, *Computer-based Medical Consultation: MYCIN*. Elsevier, New York, NY (1976).
9. U. Ahlrichs, J. Fisher, J. Denzler, C. Drexler, H. Niemann, E. Nöth and D. Paulus, Knowledge-Based Image and Speech Analysis for Service Robots, in: *Proc. IEEE Integration of Speech and Image Understanding*, Corfu, pp. 21–47 (1999).
10. S. Wermter, M. Elshaw and S. Farrand, A modular approach to self-organization of robot control based on language instruction, *Connection Sci.* **15**, 73–94 (2003).
11. http://www.w3.org/2001/sw
12. http://www.w3.org/TR/2001/NOTE-daml+oil-walkthru-20011218
13. http://www.cogsci.princeton.edu/wn
14. http://www.cogsci.princeton.edu/wn/man/lexnames.5WN.html
15. J. Hammel, K. Hall, D. Lees, L. Leifer, M. Van der Loos, I. Perkash and R. Crigler, Clinical evaluation of a desktop robotic assistant, *J. Rehabil. Res. Dev.* **26**, 1–6 (1989).

ABOUT THE AUTHORS

Rie Katsuki received the MS degree in 2002 and the PhD degree in 2005 in precision engineering from the University of Tokyo, Tokyo, Japan. From 2003 to 2004, she was a Visiting Student at the Swiss Federal Institute of Technology Lausanne (EPFL). Currently she belongs to the Corporate Research & Development Center in Toshiba Corp. Her research interests are studies for a service robot: teaching, manipulation and environmental design.

Roland Siegwart is Full Professor and Director of the Autonomous Systems Lab, Ecole Polytechnique Federale de Lausanne (EPFL). He received his ME in 1983 and his Doctoral degree in 1989 at the Swiss Federal Institute of Technology (ETH) Zurich. After his PhD studies he spent 1 year as a Post-doc at Stanford University where he was involved in micro-robots and tactile gripping. From 1991 to 1996, he worked part time as R&D Director at MECOS Traxler AG and as a Lecturer and Deputy Head at the Institute of Robotics, ETH. Since 1996 he is a Full Professor for autonomous systems and robots at the Ecole Polytechnique Federale de Lausanne (EPFL), and since 2002 also vice-dean of the School of Engineering. He is the Chairman of the recently founded Space Center at EPFL and Deputy Director of the National Center of competence in Research on Interactive Multimodal Information Management. He is heavily involved in EU projects spanning from cognitive science to intelligent cars and jet engines. He leads a research

group of around 25 people working in the field of robotics and mechatronics. He has published over 150 papers in the field of mechatronics and robotics including a textbook on mobile robotics. He is an active member of various scientific committees and co-founder of several spin-off companies. He was the General Chair of IROS 2002 and he is currently VP for Technical Activities of the IEEE Robotics and Automation Society.

Jun Ota received the PhD degree from the Faculty of Engineering, University of Tokyo in 1994. From 1989 to 1991, he joined Nippon Steel Corp. In 1991, he was a Research Associate of the University of Tokyo. In 1996, he became an Associate Professor at the Graduate School of Engineering, Uiversity of Tokyo. From 1996 to 1997, he was a Visiting Scholar at Stanford University. His research interests are multiple mobile robot systems, environmental design for robot systems, human–robot interface and cooperative control of multiple robots.

Tamio Arai received the PhD degree in Engineering from the University of Tokyo, Tokyo, Japan, in 1977. In 1987, he became a Professor in the Department of Precision Engineering, University of Tokyo. His specialties are assembly and robotics, especially multiple mobile robots, including the legged robot league of RoboCup. He contributed to the development of robot software in ISO activities. He was Director of Research into Artifacts, Center for Engineering, University of Tokyo, from 2000 to 2005 and proposed Service Engineering. He is an active member of CIRP, the Robotics Society of Japan and the Japan Society for Precision Engineering, and is Honorary President of the Japan Association for Automation Advancement.